



"Express Mail" mailing label number EV 815963400 US

Date of Deposit: June 20, 2006

Our Case No. 10022/55

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application:)
)
Shawn S. Cornelius et al.)
) Group Art Unit: 2151
Serial No.: 09/943,964)
) Examiner: Divecha, Kamal B.
Filed: August 31, 2001)
)
For: REMOTELY MONITORING A)
DATA PROCESSING SYSTEM)
VIA A COMMUNICATIONS)
NETWORK)

APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir:

This Appeal is in response to the Office Action mailed March 20, 2006¹².

06/22/2006 WASFAW1 00000016 09943964

02 FC:1402

500.00 OP

¹ Appellants are filing a Notice of Appeal concurrently with the present Appeal Brief. Since the Notice of Appeal was filed within three months of the mailing date of the Office Action and the present Appeal Brief is being filed within two months of the filing of the Notice of Appeal, the present Appeal Brief is timely filed.

² The Office Action mailed March 20, 2006 is a non-Final Office Action. Since the Office Action presents a second rejection for the claims that were pending as of March 20, 2006, Appellants have the right to appeal the rejections made in the Office Action pursuant to 37 C.F.R. § 41.31(a)(1).

I. REAL PARTY IN INTEREST

It is believed that Accenture L.L.P. is the real party of interest in this Appeal pursuant to a recorded assignment of the above-identified application to Accenture L.L. P. executed by each of the inventors of record.

II. RELATED APPEALS AND INTERFERENCES

The undersigned, John C. Freeman, is not aware of any other appeals, interferences or other judicial proceedings that may be related to, would directly affect or be directly affected by or have a bearing on the Board's decision in the pending Appeal.

III. STATUS OF CLAIMS

The status of the claims is as follows:

Claims 1-21 are rejected under 35 U.S.C. § 112, first paragraph, as failing to adequately teach how to make and use the invention, i.e., failing to provide an enabling disclosure.

Claim 22 is rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,813,634 to Ahmed.

Claims 1-12 are rejected under 35 U.S.C. § 103(a) for being obvious in

view of Ahmed and U.S. Patent No. 6,718,482 to Sato et al.

Claims 13, 16, 18 and 21 are rejected under 35 U.S.C. § 103(a) for being obvious in view of U.S. Patent Application Publication No. US 2002/0112039 to Ullman and Sato et al.

Claims 14 and 15 are rejected under 35 U.S.C. § 103(a) for being obvious in view of Ullman, Sato et al. and U.S. Patent No. 6,178,529 to Short et al.

Claim 17 is rejected under 35 U.S.C. § 103(a) for being obvious in view of Ullman, Sato et al. and U.S. Patent No. 5,428,806 to Pocrass.

Claims 19 and 20 are rejected under 35 U.S.C. § 103(a) for being obvious in view of Ullman, Sato et al. and U.S. Patent No. 6,718,482 to Hirose et al.

Claim 23 is rejected under 35 U.S.C. § 103(a) for being obvious in view of Ahmed and U.S. Patent No. 6,012,059 to Neimat et al.

The above-mentioned rejections of claims 1-23 are the subject of this Appeal.

IV. STATUS OF AMENDMENTS

A Response and two Declarations were filed on July 15, 2005 regarding a Final Office Action mailed on April 15, 2005. An Advisory Action was mailed on August 9, 2005 that indicated that the Response would be entered, but the two

Declarations would not be entered. A Request for Continued Examination was filed on September 15, 2005 that requested that the two Declarations be entered. A Final Office Action was mailed on October 26, 2005. On February 27, 2006, an Amendment was filed which amended the claims and indicated that the October 26, 2005 Office Action should not have been made Final. A non-Final Office Action was mailed on March 20, 2006, which withdrew the finality of the October 26th Office Action. No Amendment or Response has been filed regarding the March 20th Office Action prior to the filing of the present Appeal Brief.

V. SUMMARY OF CLAIMED SUBJECT MATTER

An understanding of the inventions of independent claims 1, 13 and 22 can be made upon a review of the embodiments of the inventions shown in FIGS. 1, 2 and 6-9 of the specification. Note that in the description to follow, like elements will employ identical identification numerals.

FIG. 6 shows a block diagram of an embodiment of a remote data processing system 118 that supports one or more fault analysis procedures (P. 22, ll. 7-9). The remote data processing system 118 may convey a report message (e.g., fault analysis data) to remote management system 14 (P. 22,

II. 9-11).

The primary business system 70 is coupled to base data processing system 12 (P. 22, II. 12-13). The base data processing system 12 communicates with one or more remote data processing systems 118 via the communications network 16 (P. 22, II. 13-15). Each remote data processing system 118 is coupled to the secondary business system 72 (P. 22, II. 15-16). The management system 14 monitors the base data processing system 12, the remote data processing system 118, or both (P. 22, II. 18-20).

The remote data processing system 118 of FIG. 6 includes a multi-stage, remote software module 160 (P. 22, II. 22-24). The remote status reporter 156 includes a fault detector 165 (P. 22, II. 25-26).

Remote software module 160 may include a first stage software component 161, a second stage software component 162, and a third stage software component 163 (P. 22, II. 27-30). The lines interconnecting the stages (161, 162, 163) indicate logical data paths, physical data paths, or both (P. 22, I. 30 – P. 23, I. 2). The stages (161, 162, 163) are arranged in tandem or series such that transactional data 101 or another data message received from the base data processing system 12 via the communications network 16 is processed successively by each of the stages (161, 162, 163)

or in some other defined order by one or more stages (P. 23, ll. 2-6). Once one software stage has initiated or completed processing, the data message (e.g., transactional data or a derivative thereof), is typically passed on or handed off to the next software stage, unless the remote multi-stage software module 160 is not functioning appropriately or unless the processing of the next stage is not required (P. 23, ll. 6-10).

The fault detector 165 detects whether the software module 160 is functioning appropriately by tapping into the logical data paths (or physical data paths) at logical nodes between the stages (161, 162, 163) (P. 23, ll. 11-13). If the data message (e.g., transactional data 101), its derivative, or its precursor is present at an earlier stage and absent at a later stage after the earlier stage has initiated or completed its processing, the fault detector 165 may determine that the software stage immediately following the last detected data message is at fault (P. 23, ll. 13-17).

The feedback generator 164 may be coupled to the fault detector 165 (P. 24, l. 4). When the business-to-business system is fully functional, the feedback generator 164 may recirculate a status indicator (e.g., a dummy or known sequence bit stream) that is received from the base data processing system 12 by the data receiver 172 (P. 24, ll. 5-8). If the status indicator is

present at the last stage, the feedback generator 164 may forward the status indicator or regenerate the status indicator for transmission back to base the data processing system 12 or the management system 14 (P. 24, ll. 10-13).

The status indicator may be directed to the data transmitter 174 for transmission to the base data processing system 12 as an indicator that all of the stages of the remote software module 160 are functioning and the communications network 16 is operational (P. 24, ll. 13-16).

The circulation of the status indicator flows from the base data processing system 12 to the remote data processing system 118 and then back to the base data processing system 12 for detection by the management system 14 (P. 24, 17-19). The circulation may be referred to as a heart-beat indicator because the heart-beat indicator may be configured to be present when the business-to-business system 111 (e.g., trading system) is properly operating and responsive, or alive so to speak (P. 24, ll. 19-22).

FIG. 7 shows a flow chart of a method for managing a business-to-business system (P. 26, ll. 14-15). In step S 110 of the method, a data processing system 12 or a management system 14 determines if the data processing system 12 received an incoming status message in response to a prior outgoing transactional data message communicated to a remote

data processing system (18 or 118) via a communications network 16 (P. 26, ll. 17-20). If the data processing system 12 received an incoming status message, the method continues with step S 112 (P. 26, ll. 20-22). However, if the data processing system 12 did not receive an incoming status message, the method continues with step S 114 (P. 26, ll. 22-23).

In step S 112, the method may end if the status message indicates proper receipt of an outgoing transactional message or compliance with the characteristics of a properly operating business-to-business system (P. 26, ll. 24-26).

In step 126, the management system 14 determines if one or more software modules (60 or 160) or components are functioning properly (P. 28, ll. 6-7). In the context of the remote data processing system 118 of FIG. 6, the management system determines if each software component (e.g., 161, 162, or 163) or software module 160 is operational at the remote data processing system 118 (P. 28, ll. 7-10). The software module 160 may include the first stage software component 161, the second stage software component 162, and the third stage software component 163 (P. 28, ll. 12-14). The management system 14 transmits a status message and waits for receipt of feedback associated with the status message that indicates the

status message successfully traversed one or more stages of the communications network 16 and the remote software module 160 (P. 28, ll. 14-18). If each software module or component is running properly at the remote processing system 118, the method continues with step S 134 (P. 28, ll. 20-22). If each software module or component is not running, the method continues with step S 128 (P. 28, ll. 22-23).

In step S 128, the management system 14 identifies any deficient software module or component thereof (P. 28, ll. 25-26). For example, the fault detector 165 may sense the progress of the status message to determine the identity of the software component (e.g., 161, 162, or 163) that is malfunctioning or nonoperational (P. 28, ll. 26-28).

FIG. 8 shows a flow chart of a method for monitoring a remote data processing system (18 or 118) (P. 30, ll. 5-6). In step S50, a remote data processing system (18 or 118) or a receiver 172 receives a data message via a communications network 16 (P. 30, ll. 7-8). The data message may comprise transactional data, reference data, or both for communications between a base data processing system 12 and a remote data processing system (18 or 118) (P. 30, ll. 8-10).

In step S52, the remote data processing system (18 or 118) cascades

at least a first stage software component and a second stage software component to form an installed remote software module 160 for accepting the received data message from the receiver 172 (P. 30, ll. 11-14).

In step S54, the remote data processing system (18 or 118) or the fault detector 165 detects the data message or a derivative at a group of logical nodes within the installed remote software module 160 to determine flow of the data message, or a derivative thereof, between the logical nodes (P. 30, ll. 17-20). Hence, the fault detector 165 evaluates the data flow of the data message, or a derivative thereof, through at least one of the first stage software component and the second stage software component (P. 30, ll. 20-23).

In step S56, the remote data processing system (18 or 118) or the fault detector 165 identifies a deficient software component of the installed remote software module 160 as any of said software stage components that blocks or disrupts the flow of the data message between two adjacent logical nodes (P. 31, ll. 1-4). In one example, the fault detector 165 taps into a logical data path between the stages to detect whether each of the software component stages are functioning (P. 31, ll. 4-6). During the tapping, the fault detector 165 may determine that a stage immediately following the last detected data message

is at fault (P. 31, ll. 6-8).

Contemporaneously or noncontemporaneously with the steps of the method of FIG. 8, a status code is circulated between the base data processing systems 12 and the remote data processing systems (18 or 118) over the communications networks 16 of the embodiments shown in FIGS. 1 and 6 (P. 32, ll. 1-4).

The embodiment of FIG. 6 has been previously described. Regarding the embodiment of FIG. 1, it shows a business-to-business system 10 that includes a primary business system 70 coupled to a base data processing system 12 (P. 5, ll. 4-6). The base data processing system 12 communicates with one or more remote data processing systems 18 via a communications network 16 (P. 5, ll. 6-8).

A management system 14 (e.g., monitor 36 of FIG. 2) supports remote monitoring of the following attributes of remote data processing systems 18: (1) system monitoring of one or more interactions between the base data processing system 12 and the remote data processing system 18 via the communications network 16; (2) application monitoring of application software of the base data processing system 12, the remote data processing system 18, or both (P. 8, ll. 9-14). System monitoring refers to monitoring of one or

more system components that support application software (e.g., business-to-business application software) (P. 8, ll. 14-16).

FIG. 2 shows an illustrative example of components that may be used to practice the configuration of FIG. 1. In particular, the remote data processing systems 18 may include an active remote software module 60 as shown in FIG. 2 (P. 9, ll. 3-5 and P. 14, ll. 7-9).

In the discussion to follow, the operation of the status code will be discussed with respect to the embodiments of FIGS. 1 and 6, wherein like items are identified by like numerals and alternative items are indicated in parentheses. The status code may be passed through various elements of the business-to-business system (P. 32, ll. 5-6). The status code is distinct from the status of message of FIG. 8 (P. 32, l. 6). The status code may be routed from the base data processing system 12 via the communications network 16 to a remote data processing system (18, 118) associated with the installed remote software modules (60, 160) of FIGS. 1 and 6 (P. 32, ll. 7-9). The status code is routed from the remote data processing system (18 or 118) to the base data processing system 12 via the communications network 16 to indicate that the continuity of at least one logic data path traversed by the status code (P. 32, ll. 9-12). The status code is passed from at least an input

of a communications network 16 to an output of the communications network 16 to indicate that the communications network 16 is operational (P. 32, ll. 12-15). The status code is passed from at least an input of the installed remote software module (60, 160) to an output of the installed remote software module (60, 160) to indicate that the installed remote software module (60, 160) is operational (P. 32, ll. 15-17).

FIG. 9 illustrates a method of monitoring a business-to-business system. In step S58, a base data processing system 12 transmits a status code from a base data processing system 12 to a remote data processing system (18 or 118) via a communications network 16 (P. 32, ll. 20-22). The status code traverses a first logical data path (e.g., a virtual data path or a physical data path) over the communications network 16 between the base data processing system 12 and the remote data processing system (18 or 118) if continuity of the logical data path is present (P. 32, l. 22 – P. 33, l. 3).

In step S60, a remote data processing system 118 receives the status code at a data receiver 172 in the remote data processing system (18 or 118) if continuity of the first logical data path is present (P. 33, ll. 4-6).

In step S62, the remote data processing system 118 inputs the status code into a remote software module (60, 160) of the remote data processing

system (18 or 118) (P. 33, ll. 7-9).

In step S64, the remote data processing system (18 or 118) outputs the status code from an output of the remote software module (60, 160) if the remote software module (60, 160) provides a second logical data path of continuity to the status code (P. 33, ll. 10-13).

In step S66, a transmitter 174 of the remote data processing system (18 or 118) transmits the outputted status code back to the base data processing system 12 via the communications network 16 as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment (P. 33, ll. 14-17).

Accordingly, the management system 14 may use the method of FIG. 9 to monitor end-to-end continuity of a communications path in a business-to-business system (P. 34, ll. 9-11). If the status code is sent from the base data processing system 12 or the management system 14 and not received back at the management system 14, the management system 14 is alerted to the presence of a communications fault or break in the communications path (P. 34, ll. 11-14).

There are no means-plus-function terms or step-plus-function terms in independent claims 1, 13, 22 and dependent claims 9-12, 14, 15, 17, 19 and 23, which are argued separately below in Section VII.

VI. GROUND OF REJECTION TO BE REVIEWED ON APPEAL

There are eight grounds of rejection presented for review:

1) the rejection of claims 1-21 for failing to adequately teach how to make and use the invention, i.e., failing to provide an enabling disclosure under 35 U.S.C. § 112, first paragraph;

2) the rejection of claim 22 for being anticipated under 35 U.S.C. § 102(e) in view of Ahmed;

3) the rejection of claims 1-12 for being obvious under 35 U.S.C. § 103(a) in view of Ahmed and Sato et al.;

4) the rejection of claims 13, 16, 18 and 21 for being obvious under 35 U.S.C. § 103(a) in view of Ullman and Sato et al.;

5) the rejection of claims 14 and 15 for being obvious under 35 U.S.C. § 103(a) in view of Ullman and Short et al.;

6) the rejection of claim 17 for being obvious under 35 U.S.C. § 103(a) in view of Ullman, Sato et al. and Pocrass;

7) the rejection of claims 19 and 20 for being obvious under 35 U.S.C. §

103(a) in view of Ullman, Sato et al. and Hirose et al.; and

8) the rejection of claim 23 for being obvious under 35 U.S.C. § 103(a) in view of Ahmed and Neimat et al.

VII. ARGUMENT

A. 35 U.S.C. § 112, First Paragraph

1. Claims 1-12

Claims 1-12 were rejected in the Office Action of March 20, 2006 (hereinafter “the Office Action”) under 35 U.S.C. §112, first paragraph, for failing to adequately teach how to make and use the invention, i.e., failing to provide an enabling disclosure. In particular, the Office Action asserts that there is no support in the Specification of the fault detector that detects “whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component” as recited in claim 1. The Office Action at page 4 asserts that “the specification merely suggests that the process of identifying a deficient software component of the installed remote software module as any of said software stage components that **block or disrupts** the flow of data message between two adjacent nodes (figure 8)” (emphasis as in original). Appellants traverse the rejection. Step S54 of FIG.

8 and page 30, lines 17-23 indicate that in one embodiment the fault detector 165 detects the data message or a derivative to determine the flow of data between logical nodes. Thus, the fault detector 165 inherently determines whether the data message or its derivative flows “through at least one of the first stage software component and the second stage software component” (FIG. 8, P. 30, ll. 20-23). This can be seen in FIG. 6 wherein the fault detector 165 is connected to the inputs and outputs of the first, second and third stage software components. If the fault detector 165 determines that a data message is present at an earlier stage and absent at a later stage, the fault detector 165 “may determine that the software stage immediately following the last detected data message is at fault” (P. 23, ll. 11-17). By detecting the existence of the data message at an input of a software stage, the fault detector 165 is able to determine if the data message flows entirely through the same software stage by whether or not it detects the data message at the output of the software stage. Since there is support for the phrase in question of claim 1, Appellants’ Specification adequately enables the invention of claim 1 to one of ordinary skill and reasonably conveys to one of ordinary skill in the art that Appellants had possession of the invention of claim 1. Accordingly, the rejection has no merit and so should be withdrawn.

b. Claims 13-21

Claims 13-21 were rejected in the Office Action under 35 U.S.C. §112, first paragraph, for failing to adequately teach how to make and use the invention, i.e., failing to provide an enabling disclosure. In particular, the Office Action asserts that there is no support in the Specification of the fault detector that detects “whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component.” Appellants traverse the rejection in that method claim 13 does not recite a fault detector.

If the rejection regards claim 13’s phrase “detecting the data message or a derivative . . . to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component,” the Appellants traverse the rejection for reasons similar to those given above in Section VII.A.1. Since there is support for the phrase in question of claim 13, the rejection has no merit and so should be withdrawn.

B. 35 U.S.C. § 102

Claim 22 was rejected in the Office Action under 35 U.S.C. §102(e) as

being anticipated by Ahmed. Appellants traverse the rejection for several reasons. First, claim 22 recites “outputting the status code from an output of the remote software module if the determining determines that the remote software module provides a logical data path of continuity to the status code.” Ahmed discloses having maintenance server 8 sending status query messages or “pings” to each of the computers (Col. 2, ll. 35-37). If the computers and related hardware/software are functioning properly then a reply is received from each computer (Col. 2, ll. 37-41). There is no mention in Ahmed that the reply is the same ping that was originally sent. Accordingly, claim 22 is not anticipated by Ahmed.

The rejection of claim 22 is improper for the additional reason that Ahmed fails to transmit an “outputted status code back to the base data processing system via the communications network as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment” as recited in the claim. As mentioned above, Ahmed does not disclose outputting the same ping that was originally sent. It follows that Ahmed does not disclose transmitting the same ping back to the maintenance server 8 “as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment.” Accordingly, claim 22 is not anticipated by Ahmed.

The rejection of claim 22 is improper for the additional reason that it is not relying on a single embodiment to anticipate the claims. The rejection relies on using elements from both of the embodiments of FIGS. 2 and 3 to anticipate the claims. In particular, the Office Action at pages 5 and 6 relies on the embodiment of FIG. 2 while page 6 relies on the embodiment of FIG. 3. This reliance is improper in that the Office Action is attempting to mix components from different embodiments to arrive at the claimed invention. This is improper since the elements relied in the reference must be arranged as in the claim for anticipation to be proper. *Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989).

C. 35 U.S.C. § 103

1. Ahmed and Sato et al.

a. Claims 1-8

Claims 1-8 were rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Sato et al. Appellants traverse the rejection for several reasons. First, independent claim 1 recites “a remote software module arranged to receive the data message from the data receiver, the remote software module including at least a first stage software component cascaded

with a second stage software component.” The Office Action asserts at page 7 that the PCs of FIG. 1 of Ahmed “are inherently associated with the software modules such as operating system that usually includes plurality of software stages cascaded with each other.” Appellants take this assertion to mean that Ahmed’s PCs include a remote software module “including a first software component cascaded with a second stage software component.” Appellants traverse the assertion as being improperly made under the law. To show inherency, extrinsic evidence must make it clear the missing matter is necessarily present in the reference and it would be so recognized by one of ordinary skill in the art. *In re Roberson*, 169 F. 3d 743, 49 USPQ 1949 (Fed. Cir. 1999). In the present case, there has been no extrinsic evidence provided in the Office Action showing that the PCs of FIG. 1 could not operate with only one software module or a plurality of software modules that are not cascaded with one another. Accordingly, the assertion is improper and should be withdrawn.

It is noted that the Office Action later concedes at page 7 that Ahmed does not disclose a remote software module that includes two stage software components cascaded with one another. However, the subsequent statement that Ahmed “might inherently disclose” such a remote software module is irrelevant since the Office Action has not provided extrinsic evidence to show

that such a remote software module is necessary as required by *In re Roberson*.

The rejection is improper for the additional reason that Ahmed fails to disclose “a fault detector associated with the first software stage component and the second software stage component” that detects “a fault in the remote software module by detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component” as recited in claim 1. The Office Action has asserted at page 7 that “the ping message flows through the software module in the PC (of FIG. 1) and the reply is sent” (parenthetical information added). The question is not whether the ping message flows through the software, but whether the ping message “flows entirely through at least one of the first stage software component and the second stage software component” (emphasis supplied) as recited in claim 1. There is no disclosure in Ahmed that the ping message flows entirely through a stage software component. Furthermore, there is no disclosure in Ahmed that the reply sent by its PCs of FIG. 1 is based on “detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component” as recited in claim 1. The Office Action has relied on the following passage of Ahmed as disclosing the recited fault detector:

Also shown in FIG. 1 is a connection (8) to a maintenance server (9) such as a NetView 6000 server. In this example, this connection (8) connects to the router (1) using the router's NIC for network D. The maintenance server (9) typically contains a connectivity database which contains all of the network addresses of all the elements on the other networks connected to the router, such as all the computers connected to networks A, B, and C. Using this database, the maintenance server (8) periodically sends status query messages, or "pings", to each of the computers. If each computer is on-line, the router is functioning properly, and the network physical media (cable, RF links, etc.) is in tact, a reply will be received from each computer nearly immediately in response to the "ping". If a reply or response is not received within a certain time from transmitting of the "ping", the maintenance server (9) may assume a problem with the computer, router, or network(s) exists.

For example, if all computers and the router are functioning correctly except for one computer, then only one response will not be received, and all other responses will be received. However, if the router fails, no responses will be received from any of the computers. In the most basic of maintenance system configurations such as the basic NetView 6000 product, this scenario can result in a storm of events being sent to the problem management server which correlates events and opens trouble tickets, leading to many useless and/or redundant e-mails and pagers.

FIG. 2 illustrates this scenario. A normal "ping" (20) is forwarded from the NetView 6000 to the router, which forwards (21) it to the appropriate PC. The PC, if functioning properly, replies (22) via the router to the

NetView 6000 (23) within a predetermined time limit t.sub.1. If the router has failed, the "ping" (24) will not be replied to by any of the computers within time t.sub.1, which will result in the NetView 6000 sending multiple "computer down" messages (25) to the problem management server. The problem management (Col. 2, ll. 29-64).

The above passage is silent as to either 1) having the ping messages pass entirely through a software component or 2) "detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component."

Accordingly, Ahmed does not disclose the fault detector recited in claim 1.

Sato et al. does not cure the deficiencies of Ahmed in that Sato et al. does not disclose nor suggest altering Ahmed so that it uses "a fault detector associated with the first software stage component and the second software stage component" that detects "a fault in the remote software module by detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component." In particular, Sato et al.'s embodiment of FIG. 1 includes a computer 101 that is monitored for faults taking place therein by monitoring computer 115 (Col. 4, ll. 6-9). A first operating system 105 and a second operating system 112 operate on the monitored computer 101 (Col. 4, ll. 11-13). The first operating system 105 has a fault information collecting means 106 for

collecting fault information concerning faults in the first operating system 105 when a software fault takes place in the first operating system 105 (Col. 4, ll. 31-35). The first operating system 105 detects a fault of its own and then the fault information collecting means 106 collects the fault information, such as register information and memory information during the occurrence of the fault (Col. 4, ll. 35-38). In the first operating system 105, a first fault monitor agent 104 periodically transmits an alive message 109 indicating that no fault has been detected in the first operating system 105 to the second operating system 112 (Col. 4, ll. 28-32). The alive message 109 is sent to a second fault monitor agent 108 located in the software environment of the second operating system 112 (Col. 4, ll. 28-32 and Col. 5, ll. 32-39). The second fault monitor agent 108 is in communication with a fault monitor manager 116 so as to send to the fault monitor manager 116 a fault notification from the monitored computer 101 (Col. 4, ll. 54-57). Also, the fault monitor manager 116 transmits to the second fault monitor agent 108 a command to control the monitored computer 101 (Col. 4, ll. 57-59).

The first fault monitor agent 104 transmits an alive message 109 to second fault monitor agent 108 that is indicative of normal operation of the first operating system 105. A fault detecting means 401 within the second fault monitor agent 108 decides whether the alive message 109 from the first fault

monitor agent 104 is received before a predetermined fault detection time expires (Col. 5, ll. 28-55). If the alive message 109 is not received within the fault detection time, the fault detecting means 401 determines that a fault occurs in the first operating software environment 102 (Col. 5, ll. 53-55).

Based on the above description, there are two levels of fault detection occurring. First, the first operating system 105 detects a fault within itself. There is no disclosure that such detection is performed by detecting whether a data message or a derivative thereof flows entirely through at least a software component. The second level of fault detection is determining whether an alive message 109 is received within a predetermined fault detection time by the second fault monitor agent 108. Again, such fault detection does not involve detecting whether a data message or a derivative thereof flows entirely through at least one software component.

Regarding the embodiment of FIG. 13 of Sato et al., it operates in a manner similar to that of the FIG. 1 embodiment. For example, a fault is detected by first operating system 105 itself and an alive message 109 is generated when no fault is detected by the operating system 105 (Col. 14, ll. 9-13). In addition, the alive message 109 is sent to a second fault monitor agent 108 located in the software environment of the second operating system 112, as shown in FIG. 13.

Fault monitoring board 1301 determines whether a fault takes place in the first operating system environment by checking on the value of the alive message (Col. 14, ll. 13-18). A fault detecting means 1403 within the fault monitoring board 1301 decides whether the alive message 109 from the first fault monitor agent 104 is received before a predetermined fault detection time expires. If the alive message 109 is not received within the fault detection time, the fault detecting means 1403 determines that a fault occurs in the first operating software environment 102 and causes a CPU to interrupt the system (Col. 14, ll. 31-36).

So as with the embodiment of FIG. 1, the embodiment of FIG. 13 operates by having the first operating system 105 detects a fault within itself. However, there is no disclosure that such detection is performed by detecting whether a data message or a derivative thereof flows entirely through at least a software component. In addition, the determining whether an alive message 109 is received within a predetermined fault detection time by the second fault monitor agent 108 does not involve detecting whether a data message or a derivative thereof flows entirely through at least one software component.

Since Sato et al. does not disclose nor suggest altering Ahmed to use “a fault detector associated with the first software stage component and the second software stage component” that detects “a fault in the remote software module

by detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component” the rejection is improper and should be withdrawn.

For the above reasons, the rejection of claim 1 is improper and should be withdrawn. Claims 2-8 depend directly or indirectly on claim 1 and so their rejections should be withdrawn for the same reasons stated above with respect to claim 1.

b. Claim 9

Claim 9 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Sato et al. Appellants traverse the rejection for several reasons. First, claim 9 depends indirectly on claim 1 and so is patentable over Ahmed and Sato et al. for at least the same reasons given above in Section VII.C.1.a as to why claim 1 is patentable over the references.

The rejection is improper for the additional reason that there is no motivation in either Ahmed or Sato et al. to alter Ahmed to use a fault detector that “identifies the first software stage as a faulty software component if the data message is present at an input of the first software stage, but not the output of the first software stage” as recited in claim 9. The Office Action has relied on FIG. 2 and the passage at column 2, lines 29-65 of Ahmed as disclosing the

recited fault detector. The passage is substantially presented previously in Section VII.C.1.a and it clearly does not describe the fault detector as recited in claim 9. Since Sato et al. does not suggest altering Ahmed to use a fault detector that “identifies the first software stage as a faulty software component if the data message is present at an input of the first software stage, but not the output of the first software stage”, the rejection is improper and should be withdrawn.

c. Claim 10

Claim 10 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Sato et al. Appellants traverse the rejection for several reasons. First, claim 10 depends indirectly on claim 1 and so is patentable over Ahmed and Sato et al. for at least the same reasons given previously in Section VII.C.1.a as to why claim 1 is patentable over the references.

The rejection is improper for the additional reason that there is no motivation in either Ahmed or Sato et al. to alter Ahmed to use a fault detector that “identifies the second software stage as a faulty software component if the data message is present at an input of the second software stage, but not the output of the second software stage” as recited in claim 10. The Office Action

has relied on FIG. 2 and the passage at column 2, lines 29-65 of Ahmed as disclosing the recited fault detector. The passage is substantially presented previously in Section VII.C.1.a and it clearly does not describe the fault detector as recited in claim 10. Since Sato et al. does not suggest altering Ahmed to use a fault detector that “identifies the second software stage as a faulty software component if the data message is present at an input of the second software stage, but not the output of the second software stage”, the rejection is improper and should be withdrawn.

d. Claim 11

Claim 11 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Sato et al. Appellants traverse the rejection for several reasons. First, claim 11 depends indirectly on claim 1 and so is patentable over Ahmed and Sato et al. for at least the same reasons given previously in Section VII.C.1.a as to why claim 1 is patentable over the references.

The rejection is improper for the additional reason that there is no motivation in either Ahmed or Sato et al. to alter Ahmed to use a fault detector that “identifies the first software stage as a faulty software component if a derivative of the data message is present at an input of the first software stage,

but not the output of the first software stage” as recited in claim 11. The Office Action has relied on FIG. 2 and the passage at column 2, lines 29-65 of Ahmed as disclosing the recited fault detector. The passage is substantially presented in Section VII.C.1.a and it clearly does not describe the fault detector as recited in claim 11. Since Sato et al. does not suggest altering Ahmed to use a fault detector that “identifies the first software stage as a faulty software component if a derivative of the data message is present at an input of the first software stage, but not the output of the first software stage”, the rejection is improper and should be withdrawn.

e. Claim 12

Claim 12 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Sato et al. Appellants traverse the rejection for several reasons. First, claim 12 depends indirectly on claim 1 and so is patentable over Ahmed and Sato et al. for at least the same reasons given in Section VII.C.1.a as to why claim 1 is patentable over the references.

The rejection is improper for the additional reason that there is no motivation in either Ahmed or Sato et al. to alter Ahmed to use a fault detector that “identifies the second software stage as a faulty software component if a derivative of the data message is present at an input of the second software

stage, but not the output of the second software stage” as recited in claim 12.

The Office Action has relied on FIG. 2 and the passage at column 2, lines 29-65 of Ahmed as disclosing the recited fault detector. The passage is substantially presented in Section VII.C.1.a and it clearly does not describe the fault detector as recited in claim 12. Since Sato et al. does not suggest altering Ahmed to use a fault detector that “identifies the second software stage as a faulty software component if a derivative of the data message is present at an input of the second software stage, but not the output of the second software stage”, the rejection is improper and should be withdrawn.

2. Ullman and Sato et al.

Claims 13, 16, 18 and 21 were rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ullman and Sato et al. Appellants traverse the rejection. First, independent claim 13 recites “detecting the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” The Office Action has conceded that Ullman does not disclose the recited “detecting.”

Sato et al. does not cure the deficiencies of Ullman in that Sato et al. does

not disclose nor suggest altering Ullman so that Ullman detects “the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” As mentioned in Section VII.C.1.a, Sato et al. does not disclose a fault detector that detects “a fault in the remote software module by detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component.” Accordingly, Sato et al. does not disclose nor suggest altering Ullman to perform the recited detecting. Accordingly, the rejection is improper and should be withdrawn.

For the above reasons, the rejection of claim 13 is improper and should be withdrawn. Claims 16, 18 and 21 depend directly on claim 13 and so their rejections should be withdrawn for the same reasons stated above with respect to claim 13.

3. Ullman, Sato et al. and Short et al.

a. Claim 14

Claim 14 was rejected in the Office Action under 35 U.S.C. §103 as being

obvious in view of Ullman, Sato et al. and Short et al. Appellants traverse the rejection. Claim 14 depends directly on claim 13. As pointed out in Section VII.C.2, Ullman and Sato et al. each fail to suggest altering Ullman so that Ullman detects “the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” Since Short et al. does not suggest such an alteration for Ullman either, the rejection is improper and should be withdrawn.

b. Claim 15

Claim 15 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ullman, Sato et al. and Short et al. Appellants traverse the rejection. Claim 15 depends directly on claim 13. As pointed out in Section VII.C.2, Ullman and Sato et al. each fail to suggest altering Ullman so that Ullman detects “the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” Since Short et al. does not suggest such an

alteration for Ullman either, the rejection is improper and should be withdrawn.

4. Ullman, Sato et al. and Pocrass

Claim 17 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ullman, Sato et al. and Pocrass. Appellants traverse the rejection. Claim 17 depends directly on claim 13. As pointed out in Section VII.C.2, Ullman and Sato et al. each fail to suggest altering Ullman so that Ullman detects “the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” Since Pocrass does not suggest such an alteration for Ullman either, the rejection is improper and should be withdrawn.

5. Ullman, Sato et al. and Hirosawa et al.

Claims 19 and 20 were rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ullman, Sato et al. and Hirosawa et al. Appellants traverse the rejection. Claim 19 depends directly on claim 13. As pointed out in Section VII.C.2, Ullman and Sato et al. each fail to suggest altering Ullman so that Ullman detects “the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of

the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component.” Since Hirose et al. does not suggest such an alteration for Ullman either, the rejection is improper and should be withdrawn.

For the above reasons, the rejection of claim 19 is improper and should be withdrawn. Claim 20 depends directly on claim 19 and so its rejection should be withdrawn for the same reasons stated above with respect to claim 19.


6. Ahmed and Neimat et al.

Claim 23 was rejected in the Office Action under 35 U.S.C. §103 as being obvious in view of Ahmed and Neimat et al. Appellants traverse the rejection. Claim 23 depends directly on claim 22. As pointed out in Section VII.B, Ahmed fails to disclose either 1) “outputting the status code from an output of the remote software module if the determining determines that the remote software module provides a logical data path of continuity to the status code” or 2) “transmitting the outputted status code back to the base data processing system via the communications network as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment.” Since Neimat et al. does not suggest altering Ahmed to 1) output “the status code from

an output of the remote software module if the determining determines that the remote software module provides a logical data path of continuity to the status code" and 2) transmit "the outputted status code back to the base data processing system via the communications network as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment", the rejection is improper and should be withdrawn.

For the reasons give above, Appellants respectfully submit that the rejections should be withdrawn and the claims should be allowed.

Respectfully submitted,



John C. Freeman
Registration No. 34,483
Attorney for Appellants

BRINKS HOFER
GILSON & LIONE
P.O. Box 10395
Chicago, Illinois 60610
(312) 321-4200

Dated: June 20, 2006

VIII. CLAIMS APPENDIX

1. A remote data processing system comprising:
 - a data receiver for receiving a data message;
 - a remote software module arranged to receive the data message from the data receiver, the remote software module including at least a first stage software component cascaded with a second stage software component;
 - and
 - a fault detector associated with the first software stage component and the second software stage component to detect a fault in the remote software module by detecting whether the data message or a derivative thereof flows entirely through at least one of the first stage software component and the second stage software component.
2. The system according to claim 1 further comprising a remote status reporter for reporting a status message on at least one of the remote software module and hardware of the remote data processing system.
3. The system according to claim 1 further comprising a database for storing detected faults, stage identifiers, and fault descriptions outputted by the

fault detector.

4. The system according to claim 1 further comprising a database for storing status data on corresponding components of a remote data processing system.

5. The system according to claim 1 further comprising a database, the fault detector logging one or more error messages into the database.

6. The system of claim 1 further comprising a feedback generator associated with the remote software module, the feedback generator receiving a status code outputted from the remote software module and forwarding the status code to a transmitter for transmission via a communications network.

7. The system according to claim 1 further comprising a feedback generator associated with the remote software module, the feedback generator generating a status code for a transmitter upon detection of a status code from the remote software module.

8. The system according to claim 1 wherein the fault detector has

logical connections including a connection with an input of the first software stage component, an output of the first software stage component, and an output of second software stage component.

9. The system according to claim 8 wherein the fault detector identifies the first software stage as a faulty software component if the data message is present at an input of the first software stage, but not the output of the first software stage.

10. The system according to claim 8 wherein the fault detector identifies the second software stage as a faulty software component if the data message is present at an input of the second software stage, but not the output of the second software stage.

11. The system according to claim 8 wherein the fault detector identifies the first software stage as a faulty software component if a derivative of the data message is present at an input of the first software stage, but not the output of the first software stage.

12. The system according to claim 8 wherein the fault detector identifies

the second software stage as a faulty software component if a derivative of the data message is present at an input of the second software stage, but not the output of the second software stage.

13. A method for monitoring a remote data processing system, the method comprising:

having a remote data processing system receive a data message from a base data processing system via a communications network that is external to the remote data processing system;

cascading at least a first stage software component and a second stage software component to form an installed remote software module of the remote data processing system for accepting the received data message;

detecting the data message or a derivative at a group of logical nodes within the installed remote software module to determine flow of the data message, or a derivative thereof, between the logical nodes and, hence, flow entirely through at least one of the first stage software component and the second stage software component; and

identifying a deficient software component of the installed remote software module as any of said software stage components that blocks or disrupts the flow of the data message between two adjacent logical nodes.

14. The method of claim 13 further comprising:

passing a status code, distinct from the data message, from at least an input of the installed remote software module to an output of the installed remote software module to indicate that the installed remote software module is operational.

15. The method of claim 13 further comprising:

passing a status code from at least an input of the communications network to an output of the communications network to indicate that the communications network is operational.

16. The method of claim 13 further comprising:

routing the status code from the base data processing system via the communications network to the remote data processing system; and

routing the status code from the remote data processing system to the base data processing system via the communications network to indicate that the continuity of at least one logic data path traversed by the status code.

17. The method according to claim 13 further comprising:

tapping into a logical data path between the first stage software component and the second stage software component to detect whether each of the first and second stage software components are functioning.

18. The method according to claim 13 further comprising:

determining that one of the at least a first stage software component and a second stage software component immediately following the last detected data message is at fault.

19. The method according to claim 13 further comprising:

assigning stage identifiers to distinguish the at least a first stage software component and a second stage software component from one another and to identify a faulty stage.

20. The method according to claim 19 further comprising:

associating a fault description with each of the stage identifiers for transmission to a management system via a communications network.

21. The method according to claim 13 further comprising:

archiving a fault analysis report in a database associated with the

remote processing system.

22. A method of monitoring a business-to-business system, the method comprising:

transmitting a status code from a base data processing system to a remote data processing system via a communications network;

receiving the status code at a data receiver in the remote data processing system;

inputting the status code into a remote software module of the remote data processing system;

determining whether the remote software module provides a logical data path of continuity to the status code;

outputting the status code from an output of the remote software module if the determining determines that the remote software module provides a logical data path of continuity to the status code; and

transmitting the outputted status code back to the base data processing system via the communications network as feedback indicative of the proper end-to-end continuity of communications in a business-to-business environment.

23. The method according to claim 22, the method further comprising:
storing the status code from an output of the remote software
module as a dummy transaction in the database; and
retrieving the status code as the dummy transaction in the database
and feeding the retrieved status code for transmission to the base data
processing system if the database provides a logical data path of continuity for
the status code.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.